DEVELOPING A NESTED HYDRODYNAMICAL MODEL FOR SAN DIEGO BAY, CA USING DELFT3D AND DELFTDASHBOARD

A Thesis

Presented to the

Faculty of

San Diego State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computational Science

---

by

Mohammad Abouali

Spring 2013

SAN DIEGO STATE UNIVERSITY

The Undersigned Faculty Committee Approves the

Thesis of Mohammad Abouali:

Developing a Nested Hydrodynamical Model

for San Diego Bay, CA

Using Delft3D and DelftDashboard

Jose E. Castillo, Chair
Computational Science Research Center

Barbara A. Bailey
Department of Mathematics and Statistics

Peter Blomgren
Department of Mathematics and Statistics

12/19/2012

Approval Date

Copyright © 2013
by
Mohammad Abouali
All Rights Reserved

DEDICATION

To The San Diego Community

The illiterate of the 21st century will not be those who cannot read and write, but those who cannot learn, unlearn, and relearn.

– Alvin Toffler

ABSTRACT OF THE THESIS

Developing a Nested Hydrodynamical Model
for San Diego Bay, CA
Using Delft3D and DelftDashboard
by
Mohammad Abouali
Master of Science in Computational Science
San Diego State University, 2013

In this project, three hydrodynamic models of the San Diego Bay were developed. The first one was a coarse model ($2.5\,km$ resolution) covering all of Southern California. This model affords valuable data for areas close to San Diego Bay, but is not capable of providing any information from within the bay itself. The second model is a standalone, high-resolution ($50\,m$) hydrodynamic model that is capable of simulating the hydrodynamics within the bay. However, this model is unaffected by the information from outside of the actual bay. The third is a multi-domain model of the San Diego Bay, which is a high-resolution model nested within the coarser model of Southern California. This model is capable of simulating the hydrodynamics within the bay, and enables the study of the impact of events originating well outside of the San Diego region; hence, it offers broader capabilities and applications.

All three of these models can be used to study the San Diego Bay's ecology, including the effect of toxic pollutants and the overall water quality in the region. Considering the fact that the San Diego Bay is extremely important to both the economy and ecology of the region, and taking into account that the models developed here are based on free source code (Delft3D) and freely available data sets (making the operation costs very low), it becomes fairly certain that researchers, scientists, and institutions could benefit from the models developed in this project. As a result, this model could help the San Diego community better understand the local environment, thereby enabling it to make better, more informed decisions regarding projects affecting the ecology of the region.

TABLE OF CONTENTS

PAGE

ABSTRACT vi

LIST OF FIGURES ix

ACKNOWLEDGMENTS xi

CHAPTER

1 INTRODUCTION 1

1.1 Motivation 2

1.2 Study Area 3

2 DELFT3D MODEL DESCRIPTION 6

2.1 Delft3D Grid and Coordinate System 6

2.1.1 Horizontal Grid 6

2.1.2 Vertical Grid 7

2.1.3 Staggered Grid 8

2.2 Governing Equations 9

2.2.1 Continuity Equation 9

2.2.2 Momentum Equation in Horizontal Direction 10

2.2.3 Vertical Velocity 10

2.2.4 Hydrostatic Pressure 11

2.2.5 Horizontal Stresses 11

2.2.6 Equation of State 11

2.2.7 Bed Shear Stress 12

2.3 Time and Spatial Discretization 13

3 INSTALLATION 14

3.1 Delft3D 14

3.1.1 Obtaining the Source Code 15

3.1.2 Compiling Delft3D 16

3.1.3 Testing Delft3D 17

3.1.4 Delft3D GUI 17

3.2 Delft Dashboard 18

4 SINGLE DOMAIN SAN DIEGO MODEL 20
4.1 $2.5\,km$ Resolution Southern California Model 20
4.1.1 Model Setup 20
4.1.2 Model Results 22
4.2 $50\,m$ Resolution San Diego Bay Model 28
4.2.1 Model Setup 28
4.2.2 Model Results 29
5 NESTED SAN DIEGO BAY MODEL 31
5.1 Model Setup 31
5.2 Model Results 32
6 CONCLUSION AND POSSIBLE FUTURE WORKS 34
BIBLIOGRAPHY 36

LIST OF FIGURES

PAGE

Figure 1.1. Southern California. 3

Figure 1.2. San Diego Bay. 4

Figure 1.3. San Diego Bay bathymetry constructed by combining USGS Southern California bathymetry and a private data set. 4

Figure 1.4. Approximate outline of the detailed San Diego Bay model (red line) nested inside the coarse Southern California model (yellow boundary). 5

Figure 2.1. An orthogonal curvilinear grid. Each element is addressed with a pair of (ξ, η) coordinates. source: Delft3D Example #1 - Deltares, Delft3D-FLOW User Manual..... 7

Figure 2.2. Vertical Grid: Sigma Coordinate (left) and Z-Level (right). Source: Deltares, Delft3D-FLOW user manual, Delft, the Netherlands, 2011. 7

Figure 2.3. Staggered grid used in Delft3D. Source: Deltares, Delft3D-FLOW user manual, Delft, the Netherlands, 2011. 9

Figure 3.1. DelftDashboard screen shot. 19

Figure 4.1. Southern California coarse model and control stations. The blue line shows the location of the open boundary. 21

Figure 4.2. Changes in the water level at the La Jolla station. Delft3D output (blue line) and station value (red line). 23

Figure 4.3. Changes in the water level at the Long Beach station. Delft3D output (blue line) and station value (red line). 23

Figure 4.4. Changes in the water level at the Los Angeles station. Delft3D output (blue line) and station value (red line). 24

Figure 4.5. Changes in the water level at the Newport station. Delft3D output (blue line) and station value (red line)..... 24

Figure 4.6. Changes in the water level at the Santa Barbara station. Delft3D output (blue line) and station value (red line). 25

Figure 4.7. Velocity vectors in Southern California coarse model on Feb. 1, 2008 at 00:00. 26

Figure 4.8. Velocity vectors in Southern California coarse model on Feb. 1, 2008 at 06:00. 26

Figure 4.9. Velocity vectors in Southern California coarse model on Feb. 1, 2008 at 12:00. 27

Figure 4.10. Velocity vectors in Southern California coarse model on Feb. 1, 2008 at 18:00. 27

Figure 4.11. High-resolution San Diego Bay model. 28

Figure 4.12. Changes in the water level during the first day of simulation. Delft3D output (blue line) and station value (red line). 29

Figure 4.13. Velocity vectors in high-resolution San Diego Bay Model. 30

Figure 5.1. Changes in the water level at the San Diego bay station for the nested model during the first two days of the simulation. Delft3D output (blue line) and station value (red line). 32

Figure 5.2. Changes in the water level at the San Diego Bay station. Nested San Diego Bay model (blue line) and single domain San Diego Bay model (dashed red line). 33

Figure 6.1. The San Diego Bay's entrance and structures protecting the bay. 35

ACKNOWLEDGMENTS

The best is usually kept for the last; and perhaps that's the reason that the "acknowledgements" is the last part that gets written in any thesis. I enjoy writing the "acknowledgments"; not just because I get a chance to thank people who helped me to succeed; but also because it makes me to sit back, relax for a while, and review the wonderful and amazing memories and moments that I had with them. Life, particularly nowadays, goes very fast. However, joy, beauty, love, and the good memories are "hidden away between the seconds of your life. If you don't stop for a minute, you might miss it." (Cashback, the movie, 2006).

Despite spending a very short time to prepare this thesis, there are many people whose assistance and help were absolutely necessary to make this thesis and project successful. First of all I want to thank my supervisor Prof. Jose E. Castillo, who let me to take some time off of my Ph.D. project to complete this project and obtain a second Master of Science (MS) degree. I cannot thank him enough for his assistance in this project. I want to thank my thesis committee, Prof. Peter Blomgren and Prof. Barbara Bailey, to share their time, particularly so close to the end of the year, and their inputs to make this thesis even better. I want to thank Jessica Nombrano for proof reading my thesis and the wonderful job she did. I want to thank Parisa Plant for taking care of the paper work, administration, and ordering all the computer hardware and software that I needed for this project. I also want to thank all my friends for their support and the good time we had. Particularly, I want to thank Sara and Ali for all the time we spend together and all the nights that we were staying up late working on our projects.

I also want to thank my parents, Parvin Arvaneh and Hossein Abouali, for their unconditional love. Despite being physically far away, they kept encouraging me to invest in higher educations. They have devoted their life to their children and there is no word to thank them properly. I also want to thank my sister, Azadeh Abouali, for her support even from the far distance. The last but not the least, I want to thank my girlfriend, Golnaz Badr, for her endless love and enduring support. She does a wonderful job in motivating me to proceed and advance in my life. I want to thank her for standing by me and helping me through life.

CHAPTER 1
INTRODUCTION

San Diego Bay, like all bays, provides dual usage for both commercial shipping and recreational activities. San Diego Bay is also the home of one of the largest naval bases in the United States, and a significant number of military activities occur there, including training of the US Navy SEALs (SEa, Air, Land teams). Many changes have been made to the bay since 1962, when the Port of San Diego was established to make more land available for commercial and recreational activities. For example, most of the available marshland, and more than half of all intertidal lands were reclaimed by 1975 [30]. Since the establishment of the Port of San Diego, the region has grown extensively, adding facilities such as a landmark convention center, luxury hotels, parks, and cruise and cargo terminals.

Such diverse activities do not necessarily work in favor of the region's ecology. One example being that the southern part of the San Diego Bay is covered with eelgrass, which is considered to be a very valuable shallow-water habitat that provides numerous ecological services, such as shelter, nutrient cycling, a breeding habitat for various species, stabilizing sediments, and important organic material for near-shore environments. Eelgrass requires special conditions to flourish, and is sensitive to tidal changes in the bay [13].

However, because of the extensive human activities in the region, many parts of the bay have become impaired by the presence of toxic metals and organic pollutants [39]. As a result, many institutes, organizations, and corporations are performing water quality studies in the region in order to enhance the water quality and save the natural habitat of various local birds and sea life [13, 39].

A numerical model is perhaps the best method available for use in understanding the region, and for modeling the quality of the water [39]. However, most biological and water quality models require a detailed hydrodynamic model of the region [31, 39]. Information about the hydrodynamics of a region, including knowledge of the velocity, water level, and fluxes at every grid cell, enables the researcher to perform a quantitative analysis of the water quality, and provides an understanding of the behavior of the environment, including determining the fate of different chemical compounds [8, 17, 18, 26, 27, 29].

As a result of its environmental and economic importance, there are numerous hydrodynamic studies currently underway in the region. Many measurements and experimental studies have taken place in San Diego Bay [45], including Wang's numerical hydrodynamic study dating back to 1998 [45]. Wang used a numerical grid with a $100\,m$

spatial resolution, and was able to simulate tidal water levels within an acceptable range of errors. Since then, many more numerical models have been developed, most of which have been funded or performed by the US Navy. One of the most detailed studies of the bay is that performed by the Environmental Security Technology Certification Program (ESTCP) [7]. However, the project's demonstration cost alone was approximately $580,000, which is a quarter of a million dollars more than the CH3D base model ($329,106) [7].

The majority of San Diego Bay models have either been based on curvilinear-grid hydrodynamics 3D (CH3D) or environmental fluid dynamic code (EFDC); however, almost all of them share the same CH3D grid that was originally developed by the Navy for studies of the San Diego Bay. The grid size in these models is, on average, $100\,m$, with a maximum of $250\,m$ and a minimum of slightly more than $18\,m$ [39]. Although the minimum grid length is reported to be $18\,m$, it should be noted that this only applies to one direction of the grid cells, and in those regions the cells are elongated in the opposite direction.

1.1 MOTIVATION

Due to the importance of the San Diego Bay and the vital role it plays in the local economy, it was decided to develop a high-resolution hydrodynamic model for the region. As described earlier, the outputs of the previous hydrodynamic models have played a vital role in other studies, including biogeochemical, habitat and ecological modeling studies. Numerous studies have already been performed in the region, and a few were mentioned in the previous section. However, despite many successful modeling efforts made in the past, the grid resolution of these models varies within the bay, and does not provide the same high resolution for the entire area.

Since having a high-resolution hydrodynamic model is vital for water quality and other environmental studies, it was decided to develop a high-resolution model for the San Diego Bay that would provide consistent resolution over the entire bay. As some of the past efforts have proven to be very costly (over half a million dollars) [7], our goal is to use free and open source software that can be executed on a regular desktop computer, and yet is still able to provide a consistently high-resolution model of the entire bay.

It was also decided to provide a model that is capable of creating a wetting and drying scheme. Although some of the past modeling efforts were capable of wetting and drying, a curvilinear grid was used, and only a narrow strip around the shoreline was included. The model developed for this project covers the entirety of Coronado Island, and is capable of performing both wetting and drying. (Only regions with a ground elevation exceeding $20\,m$ were set as always dry.)

1.2 Study Area

The study area in this project is limited to Southern California (Figure 1.1) and the San Diego Bay. San Diego Bay is located at roughly "$32° 39' 57'' N$" and "$117° 8' 22'' W$." It is approximately $17\,km$ long, with a maximum curved path of roughly $22\,km$. The width of the bay varies; its widest part is approximately $3.7\,km$, and it narrows down in the middle to $0.7\,km$. The San Diego Bay's mouth is about $1.8\,km$ wide, shrinking immediately down to about $0.6\,km$ (Figure 1.2).

Figure 1.1. Southern California.

The first item needed to create any ocean model is the bathymetry. Depending on the resolution of the model, the resolution of the bathymetry data set can also change. There was no public data set available that included proper bathymetry information for the interior of the San Diego Bay, as all public data sets were either too coarse to cover the bay, or the maximum depth inside the bay was erroneously shown to be a mere $1\,m$. However, we were fortunate enough to have access to high resolution bathymetry data of the interior of the bay, thanks to the high-resolution bathymetry contour lines provided to us by the US Navy. Without this data set, this project would have been impossible. The contour lines were interpolated using ILWIS and ArcGIS software to a $5\,m$ spatial resolution grid. Throughout this thesis we refer to this data set as 'SDBathy.' SDBathy was combined with the USGS's Southern California bathymetry using ArcGIS 9.3 software (SDSU License) (Figure 1.3). As can be seen in Figure 1.3, the Navy's data does not match the USGS data set at the entrance to the bay. However, as this discrepancy did not appear to have much of an impact on the hydrodynamic model, it was decided to take no action. (Later, it is shown that even with this discrepancy, our

Figure 1.2. San Diego Bay.

nested model was able to perfectly match changes in the water elevation within San Diego Bay.)

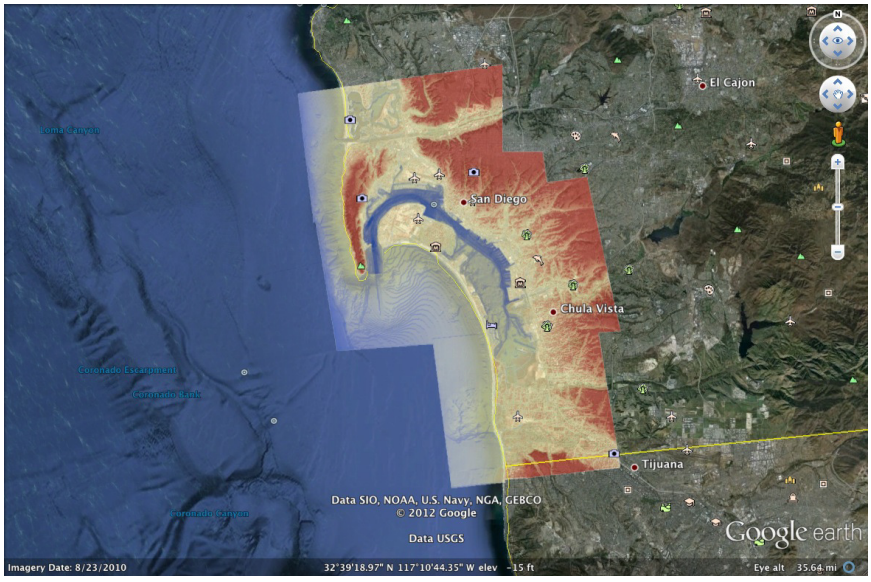

Figure 1.3. San Diego Bay bathymetry constructed by combining USGS Southern California bathymetry and a private data set.

The SDBathy data set shows that the San Diego Bay reaches a maximum depth of $72\,m$. A clear navigation channel is visible inside the bay, which is designed to provide easy access for large ships and submarines going into and out of the bay. At certain positions,

further unnatural bathymetry is seen; these positions are believed to be where submarines are located. It should be noted that San Diego Bay is home to one of the largest naval bases in the United States, and a significant amount of activity occurs there.

The goal here is to develop a high-resolution model of the San Diego Bay, nested inside a model of Southern California using Delft3D. This model should be able to predict the tidal wave and water level as precisely as possible. It should also be able to perform wet and dry computations; i.e., depending on the depth of the water, it should be able to calculate what part of the land is going to be underwater and what part will stay dry. Such schemes usually introduce a significant number of oscillations (wiggles) in the water level calculations. Therefore, we will also check if Delft3D is able to produce an oscillation-free solution. Approximate boundaries for the larger Southern California model and the detailed model of the San Diego Bay are shown in Figure 1.4. In later chapters, the full information for each grid will be provided.

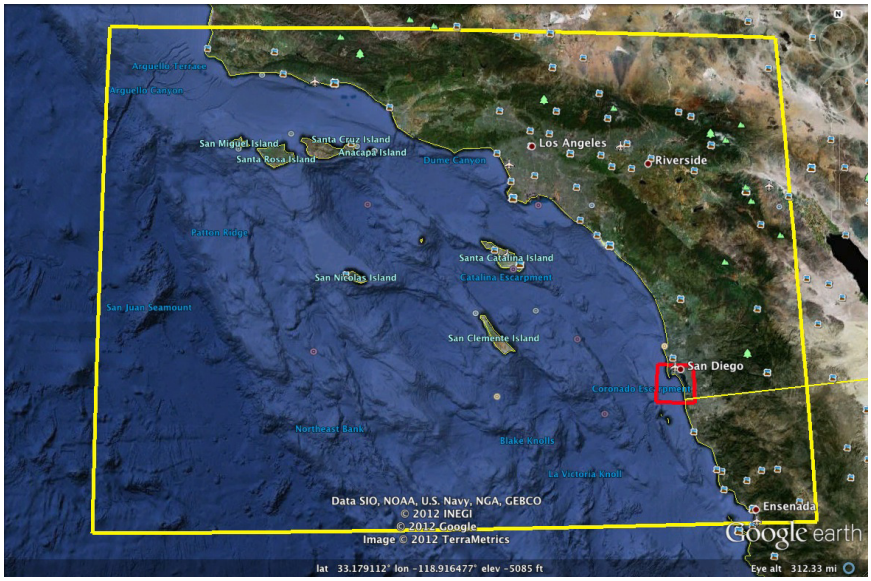

Figure 1.4. Approximate outline of the detailed San Diego Bay model (red line) nested inside the coarse Southern California model (yellow boundary).

CHAPTER 2
DELFT3D MODEL DESCRIPTION

This chapter is devoted to explaining the coordinate system, governing equations, and numerical schemes used in the Delft3D model. This chapter only focuses on those aspects of Delft3D that have been used in this project; however, Delft3D is a very comprehensive model and includes many modules and features for use in different modeling scenarios and hydraulic structures. For a full description of the model, refer to the Delft3D Flow User Manual [10].

2.1 DELFT3D GRID AND COORDINATE SYSTEM

One must first select a coordinate system in order to represent a physical space or domain. There are many choices available; ocean modeling usually requires one approach to represent the horizontal direction, and another to represent the vertical direction. The horizontal grid can affect the stability of the numerical scheme and how well the lateral boundaries are represented. However, the vertical boundary is also very important, as most of the parameterizations of the model are affected by the choice of vertical grid [2]. In this section, the different choices available in Delft3D for coordinate systems are discussed.

2.1.1 Horizontal Grid

In general, for the horizontal direction, Delft3D supports an orthogonal curvilinear coordinate system. Two options are available:

- Cartesian coordinates (ξ, η), (Figure 2.1)
- Spherical coordinates (λ, ϕ).

Rectangular and rectilinear grids are considered special cases of Cartesian coordinates. In Cartesian coordinates, the top lid of the domain is considered to be flat. In spherical coordinates λ is the longitude and ϕ is the latitude. In this coordinate, the top lid of the model follows the Earth's curvature. Spherical coordinates are also a special case of the orthogonal curvilinear grid, where:

$$\begin{aligned} \xi &= \lambda \\ \eta &= \phi \\ \sqrt{G_{\xi\xi}} &= R\cos\phi \\ \sqrt{G_{\eta\eta}} &= R \end{aligned} \tag{2.1}$$

where $R = 6378.137\,km$ is the Earth's radius, $\sqrt{G_{\xi\xi}}$, and $\sqrt{G_{\eta\eta}}$ are coefficients used to transform curvilinear coordinates into a rectangular grid.

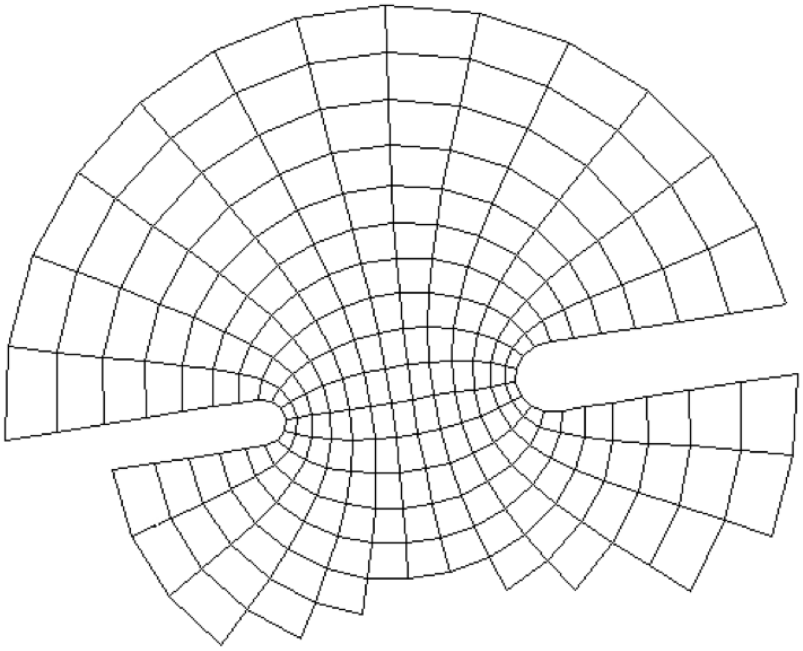

Figure 2.1. An orthogonal curvilinear grid. Each element is addressed with a pair of (ξ, η) coordinates. source: Delft3D Example #1 - Deltares, Delft3D-FLOW User Manual.

2.1.2 Vertical Grid

In the vertical direction, Delft3D offers two coordinate options, Figure 2.2, as follows:

- σ coordinate system (Sigma Coordinate),
- Z-Model or Z-Level.

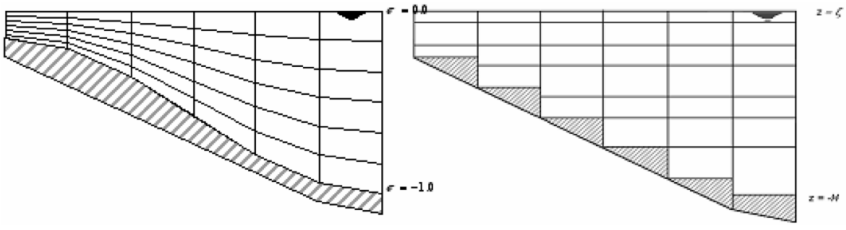

Figure 2.2. Vertical Grid: Sigma Coordinate (left) and Z-Level (right). Source: Deltares, Delft3D-FLOW user manual, Delft, the Netherlands, 2011.

Figure 2.2 illustrates how these two grids are different. The sigma coordinate was originally developed by Phillips [32], and is designed in such a way that $\sigma = -1$ at the bottom and $\sigma = 0$ at the free surface. The transformation from a physical z-coordinate to σ coordinates is done as follows:

$$\sigma = \frac{z - \zeta}{d + \zeta}, \tag{2.2}$$

where:

- ζ is the free surface elevation above the reference plane.
- d is the the depth below the reference plane.

It should be noted that the sigma coordinate was originally developed for slopes up to $45°$ [19, 20]. Slopes steeper than that will produce numerical errors, and it has been shown that over very steep slopes, sigma-coordinates produce poor results [1, 5, 15, 16, 28]. Partial derivatives can be calculated using the chain rule of derivation, which will then introduce some additional terms [1, 38].

In coastal seas, estuaries, lakes, and generally in places where there is steep topography or bathymetry, the sigma coordinate can produce numerical errors. The slope is also a function of the horizontal grid resolution. In coarse resolutions, the bottom bathymetry is represented smoothly, and most of the high-frequency variations in the bathymetry are filtered out [1]. The sigma coordinate, despite being a boundary-fitted coordinate, does not necessarily have enough resolution around the pycnocline [23, 25, 38]. One approach to overcoming this issue is to use curvilinear coordinates in the vertical direction [1]. Another common approach, which is also supported by Delft3D, is to use the Z-level (Z-grid) in the vertical direction. In the Z-grid, the horizontal lines nearly match those of the isopycnal's lines; i.e., they are parallel to the density interfaces.

2.1.3 Staggered Grid

In any hydrodynamic model, including in Delft3D, there are several variables that are simulated, including three components of the velocity, the pressure, salinity, and temperature. In general, they can be divided into vector variables, such as velocity; and scalar variables, such as pressure, salinity, and temperature. Depending on how these different variables are arranged in a grid, one can have different grid types, known as type A, B, C, D, and E [15]. Delft3D uses a C-grid, also known as a staggered grid (Figure 2.3). In a staggered grid, the scalar values are stored at the cell center, and different components of the vector variables (usually the velocity) are stored at the middle of the cell faces.

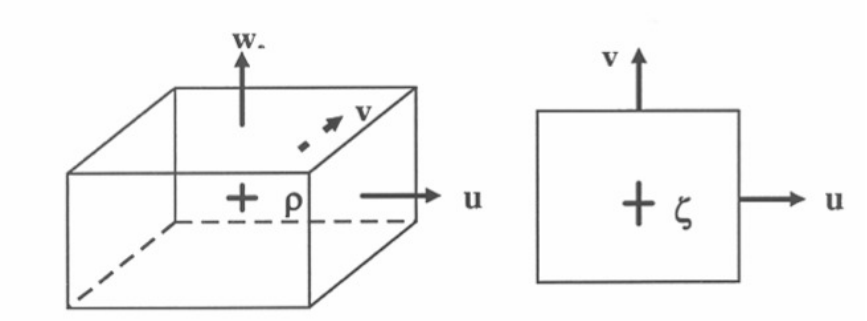

Figure 2.3. Staggered grid used in Delft3D. Source: Deltares, Delft3D-FLOW user manual, Delft, the Netherlands, 2011.

2.2 GOVERNING EQUATIONS

Delft3D uses nonlinear shallow water equations in 2D and 3D. Shallow water equations (SWE) are derived by averaging the full Navier-Stokes equation in the vertical direction. Several assumptions have been made to derive these equations: the main assumption is that the horizontal length scale is much larger than the vertical length scale. This is normally true for any ocean flow model. However, this assumption practically reduces the vertical momentum equation to a hydrostatic pressure equation. While this is a valid assumption in coarse resolution, extra care must be taken in very fine resolution cases, as well as those where fluid flow interaction with the bottom bathymetry is the dominant process [1]. In these regions, vertical velocity plays an important role in mixing, and even in the carrying of the energy [1, 4, 9, 14, 21, 33, 40, 41, 42].

Despite some of the limitations of nonlinear shallow water equations, they can be still used for many applications. In fact, the majority of ocean models currently available use this set of equations. The following reviews only the most important set of equations in Delft3D, which were used in this project. As mentioned before, Delft3D has many other modules and capabilities. For further information, including the governing equations for those modules, refer to the Delft3D Flow User Manual [10].

2.2.1 Continuity Equation

The depth-averaged continuity equation with source and sink terms in Delft3D is written as follows:

$$\frac{\partial \zeta}{\partial t} + \frac{1}{\sqrt{G_{\xi\xi}}\sqrt{G_{\eta\eta}}} \frac{\partial \left[(d+\zeta)U\sqrt{G_{\eta\eta}}\right]}{\partial \xi} + \frac{1}{\sqrt{G_{\xi\xi}}\sqrt{G_{\eta\eta}}} \frac{\partial \left[(d+\zeta)V\sqrt{G_{\xi\xi}}\right]}{\partial \eta} = Q, \qquad (2.3)$$

where:

- $\sqrt{G_{\xi\xi}}$, and $\sqrt{G_{\eta\eta}}$ are coefficients used to transform curvilinear coordinates to rectangular grid,

- U and V are the depth-integrated velocity in computational domain.
- Q is the source/sink term which is defined as follows:

$$Q = H \int_{-1}^{0} (q_{in} - q_{out})\, d\sigma + P - E. \tag{2.4}$$

In Equation 2.4, $H = d + \zeta$, P is the precipitation, E is the evaporation, q_{in} is any source of water, and q_{out} is any sink for water (such as an intake of power plant for its cooling system).

2.2.2 Momentum Equation in Horizontal Direction

The momentum equation can be written as:

$$\begin{aligned}\frac{\partial u}{\partial t} = & -\frac{u}{\sqrt{G_{\xi\xi}}}\frac{\partial u}{\partial \xi} - \frac{v}{\sqrt{G_{\eta\eta}}}\frac{\partial u}{\partial \eta} - \frac{\omega}{d+\zeta}\frac{u}{\partial \sigma} \\ & + \frac{v^2}{\sqrt{G_{\xi\xi}}\sqrt{G_{\eta\eta}}}\frac{\partial \sqrt{G_{\eta\eta}}}{\partial \xi} - \frac{uv}{\sqrt{G_{\xi\xi}}\sqrt{G_{\eta\eta}}}\frac{\partial \sqrt{G_{\xi\xi}}}{\partial \eta} \\ & + fv + F_\xi + M_\xi \\ & - \frac{1}{\rho_0\sqrt{G_{\xi\xi}}}P_\xi + \frac{1}{(d+\zeta)^2}\frac{\partial}{\partial \sigma}\left(\nu_V \frac{\partial u}{\partial \sigma}\right)\end{aligned} \tag{2.5}$$

and

$$\begin{aligned}\frac{\partial v}{\partial t} = & -\frac{u}{\sqrt{G_{\xi\xi}}}\frac{\partial v}{\partial \xi} - \frac{v}{\sqrt{G_{\eta\eta}}}\frac{\partial v}{\partial \eta} - \frac{\omega}{d+\zeta}\frac{v}{\partial \sigma} \\ & - \frac{uv}{\sqrt{G_{\xi\xi}}\sqrt{G_{\eta\eta}}}\frac{\partial \sqrt{G_{\eta\eta}}}{\partial \xi} + \frac{u^2}{\sqrt{G_{\xi\xi}}\sqrt{G_{\eta\eta}}}\frac{\partial \sqrt{G_{\xi\xi}}}{\partial \eta} \\ & - fu + F_\eta + M_\eta \\ & - \frac{1}{\rho_0\sqrt{G_{\eta\eta}}}P_\eta + \frac{1}{(d+\zeta)^2}\frac{\partial}{\partial \sigma}\left(\nu_V \frac{\partial v}{\partial \sigma}\right),\end{aligned} \tag{2.6}$$

where u and v are the eastward and northward velocity in the physical domain, respectively; ν_V is the vertical eddy viscosity coefficient [34]; F_ξ and F_η represent the unbalance of the horizontal Reynold's stresses; M_ξ and M_η represent the contribution due to external sources or sinks; and f is the Coriolis coefficient.

2.2.3 Vertical Velocity

Remember that ω in Equations 2.5 and 2.6 is the vertical velocity relative to the sigma Plane, and that it is not the vertical velocity in the physical domain. ω is calculated from the

continuity equation; the vertical physical velocity is calculated (only for post-processing purposes), as follows:

$$w = \omega + \frac{1}{\sqrt{G_{\xi\xi}}\sqrt{G_{\eta\eta}}}\left[u\sqrt{G_{\eta\eta}}\left(\sigma\frac{\partial H}{\partial \xi}+\frac{\partial \zeta}{\partial \xi}\right)+v\sqrt{G_{\xi\xi}}\left(\sigma\frac{\partial H}{\partial \eta}+\frac{\partial \zeta}{\partial \eta}\right)\right] + \left(\sigma\frac{\partial H}{\partial t}+\frac{\zeta}{\partial t}\right). \tag{2.7}$$

2.2.4 Hydrostatic Pressure

Remember that in Equation 2.5 and 2.6, pressure is defined as hydrostatic pressure. Therefore, in case of constant density, the pressure terms in the momentum equation are:

$$\frac{1}{\rho_0\sqrt{G_{\xi\xi}}}P_\xi = \frac{g}{\sqrt{G_{\xi\xi}}}\frac{\partial \zeta}{\partial \xi}+\frac{1}{\rho_0\sqrt{G_{\xi\xi}}}\frac{\partial P_{atm}}{\partial \xi}, \tag{2.8}$$

$$\frac{1}{\rho_0\sqrt{G_{\eta\eta}}}P_\eta = \frac{g}{\sqrt{G_{\eta\eta}}}\frac{\partial \zeta}{\partial \eta}+\frac{1}{\rho_0\sqrt{G_{\eta\eta}}}\frac{\partial P_{atm}}{\partial \eta}. \tag{2.9}$$

If the density is not constant, one gets:

$$\frac{1}{\rho_0\sqrt{G_{\xi\xi}}}P_\xi = \frac{g}{\sqrt{G_{\xi\xi}}}\frac{\partial \zeta}{\partial \xi}+g\frac{d+\zeta}{\rho_0\sqrt{G_{\xi\xi}}}\int_\sigma^0\left(\frac{\partial \rho}{\partial \xi}+\frac{\partial \rho}{\partial \sigma}\frac{\partial \sigma}{\partial \xi}\right)\,d\sigma', \tag{2.10}$$

$$\frac{1}{\rho_0\sqrt{G_{\eta\eta}}}P_\eta = \frac{g}{\sqrt{G_{\eta\eta}}}\frac{\partial \zeta}{\partial \eta}+g\frac{d+\zeta}{\rho_0\sqrt{G_{\eta\eta}}}\int_\sigma^0\left(\frac{\partial \rho}{\partial \eta}+\frac{\partial \rho}{\partial \sigma}\frac{\partial \sigma}{\partial \eta}\right)\,d\sigma'. \tag{2.11}$$

2.2.5 Horizontal Stresses

The horizontal stresses in the momentum equation can be reduced to a Laplace's operator [3, 6, 38] as follow:

$$F_\xi = \nu_H\left(\frac{1}{\sqrt{G_{\xi\xi}}\sqrt{G_{\eta\eta}}}\frac{\partial^2 u}{\partial \xi^2}+\frac{1}{\sqrt{G_{\xi\xi}}\sqrt{G_{\eta\eta}}}\frac{\partial^2 u}{\partial \eta^2}\right), \tag{2.12}$$

$$F_\eta = \nu_H\left(\frac{1}{\sqrt{G_{\xi\xi}}\sqrt{G_{\eta\eta}}}\frac{\partial^2 v}{\partial \xi^2}+\frac{1}{\sqrt{G_{\xi\xi}}\sqrt{G_{\eta\eta}}}\frac{\partial^2 v}{\partial \eta^2}\right). \tag{2.13}$$

2.2.6 Equation of State

The equation of state (EOS) determines the water density (ρ) as a function of the salinity and Temperature, and features different formulations for seawater. Delft3D supports both Eckart's equation [12] and the UNESCO equation [43, 44]. Eckart's equation has several limitations; however, the UNESCO equation (also called EOS80) has proven to have better

performance ($3.6\,g/m^3$ error). Delft3D uses EOS80 by default; EOS80 can be written as follows:

$$\rho = \rho_0 + As + Bs^{(3/2)} + Cs^2, \tag{2.14}$$

where:

$$\begin{aligned}
\rho_0 = \quad & 999.842594 + 6.793952 \cdot 10^{-2}t - 9.095290 \cdot 10^{-3}t^2 && (2.15)\\
& +1.001685 \cdot 10^{-4}t^3 - 1.120083 \cdot 10^{-6}t^4 + 6.536332 \cdot 10^{-9}t^5, \\
A = \quad & 8.24493 \cdot 10^{-1} - 4.0899 \cdot 10^{-3}t + 7.6438 \cdot 10^{-5}t^2 && (2.16)\\
& -8.2467 \cdot 10^{-7}t^3 + 5.3875 \cdot 10^{-9}t^4, \\
B = \quad & -5.72466 \cdot 10^{-3} + 1.0227 \cdot 10^{-4}t - 1.6546 \cdot 10^{-6}t^2, && (2.17)\\
C = \quad & 4.8314 \cdot 10^{-4}. && (2.18)
\end{aligned}$$

EOS80 is valid for $t \in [0\,^\circ C, 40\,^\circ C]$ and $s \in [0.5\,ppt, 43\,ppt]$, where "ppt" stands for "part per thousands". It should be noted that, in the presence of other chemicals in the ocean water, the above mentioned equation may no longer hold true.

2.2.7 Bed Shear Stress

Delft3D uses the logarithmic law of the wall for 3D models to calculate the bed shear stress. However, in 2D models, including for this project, the quadratic friction law is used. The quadratic friction law can be written as follows:

$$\vec{\tau} = \frac{\rho_0 g \vec{U} \left|\vec{U}\right|}{C_{2D}^2}, \tag{2.19}$$

where $\left|\vec{U}\right|$ is the magnitude of the horizontal velocity. C_{2D} is the 2D Chézy coefficient and Delft3D provides the following options:

- **Chézy Formulation**, user defined value in $[m^{1/2}/s]$.
- **Manning's Formulation:**

  $$C_{2D} = \frac{H^{1/6}}{n}, \tag{2.20}$$

  with n being the Manning's coefficients in $[m^{-1/3}s]$.
- **White Colebrook's formulation:**

  $$C_{2D} = 18 \log_{10}\left(\frac{12H}{\kappa_s}\right), \tag{2.21}$$

  with κ_s being the Nikuradse roughness length.

2.3 TIME AND SPATIAL DISCRETIZATION

Delft3D-FLOW uses the alternating direction implicit (ADI) method, as described by Leendertse [22, 24, 25] to integrate shallow water equations in time. Delft3D-FLOW uses three different spatial discretizations. In all cases, the discretization is at least second-order accurate in space. These schemes are:

- WAQUA-Scheme, [35, 37].
- Cyclic Method, [37].
- Flooding Scheme, [36].

Neither the WAQUA scheme nor the Cyclic scheme impose any time step restrictions, and are both high-order schemes. The flooding scheme is suitable for problems including rapidly varying flows, such as in hydraulic jumps. This scheme, however, imposes a time step restriction by the Courant number for advection. For further details please see the above mentioned references.

CHAPTER 3
INSTALLATION

In this chapter, the general outline of installing both Delft3D and Delft DashBoard (DDB) is explained. Delft3D is available on both Linux and Windows machines. Both Delft3D and DDB are open source and are available freely for download. The prerequisite packages and software will be also listed in this chapter.

3.1 DELFT3D

Delft3D is a modular open source code developed by Deltares, and provides an integrated framework for a multi-disciplinary approach to creating 3D computer simulations for rivers, lakes, and coastal and estuarine areas [11]. Despite its name, Delft3D is capable of simulation in both 3D and 2D; in 2D cases, shallow water equations are solved. Shallow water equations are derived by integrating Navier-Stokes equations in the vertical direction. Delft3D can be used in various areas of application, such as [10]:

- Tide- and wind-driven flows (i.e., storm surges)
- Stratified and density-driven flows
- River flow simulations
- Simulations in deep lakes and reservoirs
- Simulations of tsunamis, hydraulic jumps, bores, and flood waves
- Freshwater river discharges into bays
- Salt intrusion
- Thermal stratification in lakes, seas, and reservoirs
- Cooling water intakes and wastewater outlets
- Transport of dissolved material and pollutants
- Online sediment transport and morphology
- Wave-driven currents
- Non-hydrostatic flows

Delft3D can handle rectangular, rectilinear, and curvilinear grids. However, in the vertical direction, it supports only sigma-coordinates and Z-levels. Delft3D is a modular code, and can be coupled with other models, such as ecological and biological models. There are

many utilities developed for Delft3D in order to facilitate both the preprocessing and post-processing steps of a simulation task. The followings are a subset of these utilities, which are widely used.

- **Delft3D-RGFGRID:** This tool can be used to generate curvilinear grids.
- **Delft3D-QUICKIN:** This tool can be used to manipulate grid-oriented data, such as bathymetry or initial conditions.
- **Delft3D-NESTHD:** This tool can be used to generate boundary conditions while nesting two different models.
- **Delft3D-QUICKPLOT:** This tool, which is mainly generated using MATLAB ©, can be used to visualize the model output.
- **Delft DashBoard:** This tool is also developed in MATLAB ©, and can be used for the preprocessing step. Delft DashBoard has access to many online databases and facilitates the setup of a model.
- **OpenDA:** Originally developed in Java, this tool can be used for data assimilation using Delft3D and other programs that support the OpenDA standard.

3.1.1 Obtaining the Source Code

Deltares decided to make the full source code of Delft3D-FLOW (including the morphology) and Delft3D WAVE Engines available to the public under GPLv3 conditions. In order to download the code, one must first create a free user account. The source code, installation procedure, and manual are available at:

http://oss.deltares.nl/web/opendelft3d/source-code

The above link can also be used to create the free user account required to download the full source code. To download the code, one needs to make use of a version-control software, for example SubVersioN (SVN) on Linux machines. The Delft3D repository contains several branches; however, the fully tested and stable version of the code can be found in the ”tags” folder. The latest edition is the one with the highest version number. At the time of this writing, the latest version was 5.00.10.1983. To check for the latest version, go to the following address:

https://svn.oss.deltares.nl/repos/delft3d/tags/

If you are using a command line version-control program, such as SVN, you can check for the latest version by issuing the following command:

```
svn checkout https://svn.oss.deltares.nl/repos/ &
                delft3d/tags/5.00.10.1983 delft3d_repository
```

Notice again that 5.00.10.1983 was the latest version as of this writing. Under the Windows operating system, you can make use of a graphical version-control program such as ”TortoiseSVN,” which is freely available for download at:

http://www.http://tortoisesvn.net

You can download TortoiseSVN for both 32-bits and 64-bits Windows operating systems.

3.1.2 Compiling Delft3D

Delft3D has been tested and compiled on both Linux and Windows machines. The following section explains how to compile the source code on each system.

3.1.2.1 WINDOWS MACHINE

To compile the Delft3D source code on a Windows platform, the following software must be installed:

- Visual Studio 2008 (VS2008), or Visual Studio 2010 (VS2010)
- Intel FORTRAN compiler Version 11.0 or Version 12.0

Note that Visual Studio is freely available for students and educational purposes. However, the student version of the Intel compiler is available for a moderate cost. Recently, the Intel FORTRAN compiler version 13.0 was also supported; however, in this project, VS2008 and the Intel FORTRAN compiler version 11.0 were used to compile the code.

Once you have installed the above mentioned software, you must open the project file in either VS2008 or VS2010. All that is needed then is to choose the ”build” button or to press ”Control + Shift + B” key combination. Make sure that the ”release” version is selected in the compile options. Depending on your system, the compile procedure may take some time.

3.1.2.2 LINUX MACHINES

According to Deltares, the following packages are needed on a Linux machine prior to starting the compile procedure:

- GNU Auto Tools
- GNU Lib Tools
- GNU C++ Compiler
- expat-devel
- GNU FORTRAN Compiler
- Mpich2

- Lex
- Yacc
- OpenSSL
- Readline-devel
- Ruby Interpreter

Some of the above mentioned packages may have already been installed already by default on your Linux platform. All of these software packages are freely available. It is advised to install them using the package manager of your Linux distribution, such as ”apt-get” on Ubuntu Linux and ”Yum” on Fedora Linux. You can replace ”GNU FORTRAN compiler” with Intel FORTRAN compiler. Intel FORTRAN compiler provides faster binaries. Unlike for Windows machines, Intel FORTRAN compiler is available free of charge for personal use on the Linux environment.

Once the prerequisite packages are installed, you can compile the source code. The best option is to open ”autogen.sh” under the ”src” folder, and edit the variables to match them to those of the system. One can control the compiler, compiler flags, and other options that are needed to successfully compile the code on a Linux machine. Once the proper changes have been made, one needs to execute ”autogen.sh.” Instead of changing ”autogen.sh,” it is easier to pass all the common variables at the command line. For example, to set compiler options, try:

```
./autogen.sh CFLAGS='-O2 -m64 -fPIC'
```

”autogen.sh” will create the required ”Makefile” on your system. Now, to fully compile the source code and obtain the binaries, one needs to issue:

```
make ds-install
```

3.1.3 Testing Delft3D

Once you have fully compiled the Delft3D source code, it is advised to run some of the examples that are provided in the ”examples” folder. These examples should run without generating any errors. If you are able to run these examples, your compile procedure was successful.

3.1.4 Delft3D GUI

So far, how to obtain the Delft3D Flow source code and compile it has been discussed. By successfully compiling the code, you will have access to the solver part of Delft3D. There is a Graphical User Interface (GUI) available for Delft3D-FLOW, which facilitates setup of the model, changing parameters, and visualizing results. Deltares is planning to make the

source code for Delft3D GUI (known as Delft3d Menu) publicly available. However, this promise has not yet materialized. Meanwhile, Deltares is providing the binaries for their GUI on Windows and Linux machines. To obtain the GUI binaries and free license, send an e-mail to:

sales@deltaressystems.nl

3.2 DELFT DASHBOARD

Delft Dashboard (DDB) is part of Open Earth Tools, and is a stand-alone MATLAB-based software. DDB provides a graphical user interface that supports the modeler in the setting up of a new model, or in altering an existing model. It is currently fully integrated with Delft3D FLOW. DDB provides easy access to many online databases through ”Open-source Project for a Network Data Access Protocol” (OPeNDAP). One can access different measuring stations, such as those of the International Hydrographic Organization (IHO), or the XTide Tidal Stations. DDB also provides easy access to various bathymetry databases, including:

- GEneral Bathymetry Chart of the Ocean (GEBCO)
- National Geophysical Data Center (NGDC) Coastal Relief Model
- Shuttle Radar Topography Mission (SRTM) v4.1 (Only Land data)
- United State Geological Survey (USGS) - Hawaii
- USGS - San Francisco Bay
- USGS Southern California
- Rijks Water Staat
- Southeastern Universities Research Association (SURA) - Gulf of Mexico
- European Marine Observation and Data Network (EMODnet) - Adriatic Sea - Ionian Sea - Central Mediterranean
- EMODnet - Aegean Sea- Levantine Sea
- EMODnet - Bay of Biscay - Iberian Coast
- EMODnet - Celtic Seas
- EMODnet - Greater North Sea
- EMODnet - Western Mediterranean
- Marine Scotland - West of Lewis

The user can also import his/her own bathymetry into the DDB. It should be noted that, in this project, we made use of a very high-resolution bathymetry data that was made

available to us for the San Diego Bay. Outside of the bay, we made use of the GEBCO bathymetry data. A screen shot of Delft Dashboard is shown in Figure 3.1.

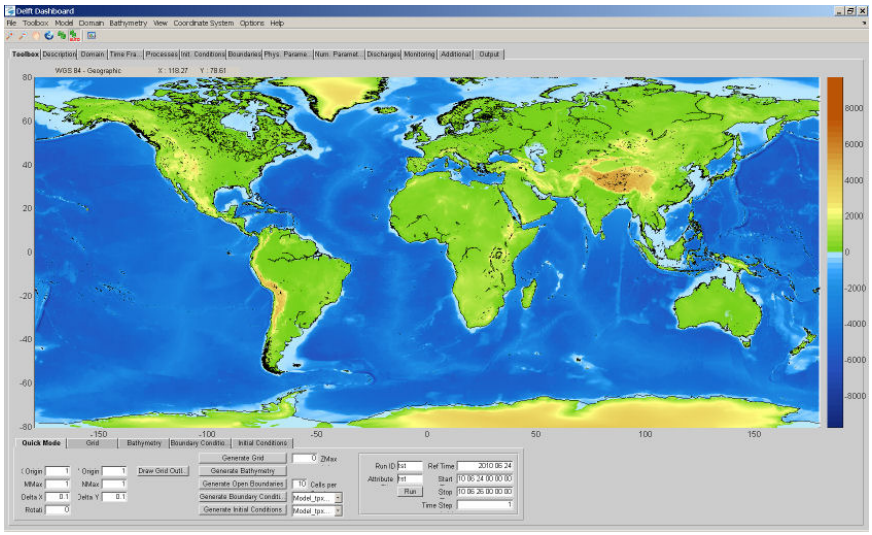

Figure 3.1. DelftDashboard screen shot.

The Delft Dashboard (DDB) binaries can be downloaded from the following link:

https://publicwiki.deltares.nl/display/OET/DelftDashboard

Once the binaries are obtained, the installation is fairly straightforward. Although the DDB is MATLAB-based, one does not need to have MATLAB installed. However, MATLAB runtime libraries, which are freely made available by Mathworks, must be installed. Instead of the DDB binaries, one may decide to download the MATLAB source code for the DDB and use the software directly through MATLAB. To do this, first download The Open Earth Tool (OET). Although we utilized the MATLAB source code of the DDB (and not the binaries) for this project, we omit the instruction on how to download the OET.

Still, the instructions can be found in the Delft Dashboard manual at:

http://publicwiki.deltares.nl/display/ddb/Download

CHAPTER 4

SINGLE DOMAIN SAN DIEGO MODEL

Both coarse and fine resolution models are initially setup using Delft Dashboard (DDB). Delft3D GUI was later used to fine-tune the model, including choosing a stable time step and setting the bed shear parameters.

4.1 $2.5\,km$ RESOLUTION SOUTHERN CALIFORNIA MODEL

As mentioned earlier, two single domain model were developed. This section explains setting up the coarse resolution model,i.e. $2.5 \times 2.5\,km$. First it is explained how the model was setup; followed by discussing the model output results.

4.1.1 Model Setup

Before setting up the model using DDB, one has to decide on the coordinate system. Since, the focus of this project is Southern California and San Diego, it was decided to use the Universal Transverse Mercator (UTM) coordinate system. UTM zone 11N covers Southern California and San Diego Bay. World Geodetic System 84 (WGS84) is used as the Datum. The domain was spatially discretized using $dx = dy = 2500\,m$. 145 grid points where selected in the M-direction (i.e., the eastward direction), and 122 grid points were selected in the N-direction (i.e., the northward direction). Therefore, the model covers an area of $362.5\,km \times 305.0\,km$ (Figure 4.1).

Those parts of the land with an elevation of $20\,m$ or higher were set to be always dry. Any parts of the land with an elevation lower than $20\,m$ have the potential to become either wet or dry. Most of the southern and eastern parts of the domain are treated as open boundaries. The reflection factor was set to a high number in order to prevent any reflection of waves back into the domain, so that all waves can freely exit the domain. Only one layer in the vertical direction was selected; making the model a true shallow water equation model. Only GEBCO bathymetry is interpolated onto the grid.

The simulation start time was set at January 1, 2008, with a full six months to be simulated (the stop time being set for midnight on July 1, 2008). The stable time step, using the ADI scheme, was found to be 1 minute. The water level for the entire domain was set at $0\,m$.

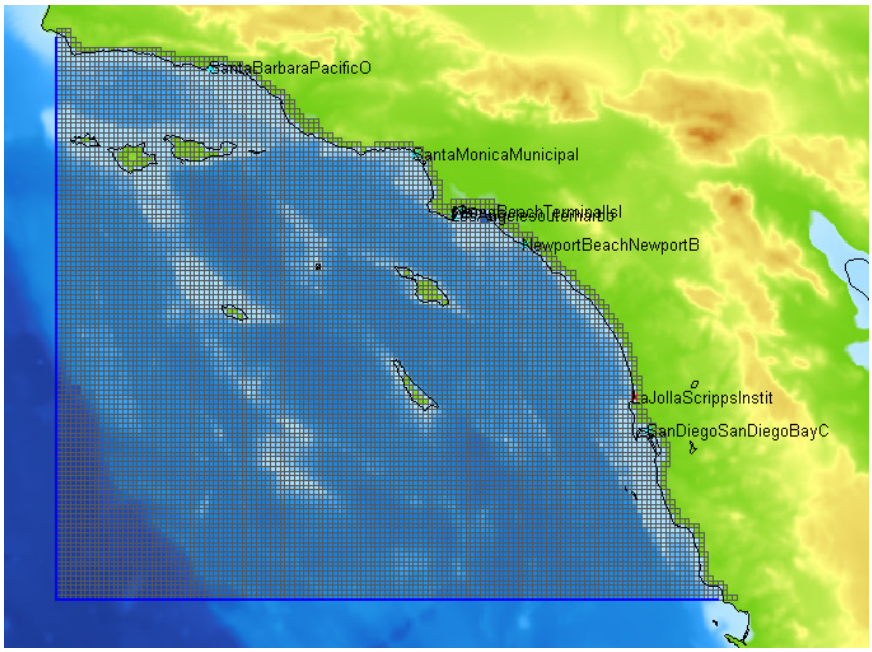

Figure 4.1. Southern California coarse model and control stations. The blue line shows the location of the open boundary.

Over land, no flow condition is enforced. However, several options are available for the open boundaries, as follows:

- **Astronomic**: The flow conditions are specified using tidal constituents, amplitude and phases.
- **Harmonic:** The flow conditions are specified using user-defined frequencies, amplitudes, and phases.
- **QH-relation**: The water level is derived from the computed discharge leaving the domain through the boundary.
- **Time Series:** The flow conditions are specified as time series.

Since we were interested in tidal simulation, in this model the astronomic boundary condition was selected and the water level at the open boundary was defined. It should be noted that, over the open boundary, each set of 10 cells were grouped together as a single boundary, making it possible to change the boundary condition on a specified part of the domain. However, it was decided to set the same boundary condition over the entire open boundary.

The Earth's gravity was set to $9.81 m/s^2$ and the water density was set at $1024 kg/m^3$. The Manning formulation was chosen for the bottom friction.

Seven monitoring points were set as observation points and then introduced into the model. Their names and locations (in grid coordinates) are listed below:

- **Santa Barbara, Pacific Ocean** (34,113)
- **Santa Monica, Municipal Pier** (77,95)
- **Los Angeles, Outer Harbor** (85,82)
- **Long Beach, Terminal Island** (87,83)
- **Newport Beach, Newport Bay Entrance, Corona del Mar** (100,76)
- **La Jolla Scripps Institute** (123,44)
- **San Diego, San Diego Bay** (126,37)

These monitoring stations are used to control the performance of the model.

It was decided to record all model outputs every 30 minutes at each of the monitoring stations. This output was stored in a file known as the history file in Delft3D. Furthermore, every 6 hours, the entire model output at each of the locations was stored on the hard disk. This is known as the map file in Delft3D, and is the largest output file. The total storage for the map file was more than 650 MB. It was also decided to store a restart file every 30 days.

4.1.2 Model Results

The low-resolution Southern California model took slightly more than 4 hours to simulate a 6-month time period (on an Intel i3 system it takes about 2 hours). The model started at a zero water level everywhere, but gradually began to adapt itself to the changes and the real profile of the water level. The water level comparison is shown in Figure 4.2, Figure 4.3, Figure 4.4, Figure 4.5, and Figure 4.6 for the tide stations in La Jolla, Long Beach, Los Angeles, Newport Beach, and Santa Barbara, respectively. The blue line is the Delft3D output and the red line is the tide station output. As can be seen, Delft3D consistently under predicts the extremums of the water level by only few centimeters; however, this is within acceptable range. However, Delft3D was completely unsuccessful in estimating a proper water level for the San Diego Bay and Santa Monica stations. The calculated water level is equal to zero for both stations throughout the entire simulation time.

The San Diego station is located well inside the bay, and neither the grid resolution nor the accuracy of the GEBCO bathymetry is capable of representing the location of the station properly. Therefore, it was not a surprise to get poor results for the San Diego station. In fact, this result was expected. However, the problem with the Santa Monica station is due to the inaccuracies in the GEBCO bathymetry data close to the coast.

The maximum water level reached only $1\,m$; therefore, throughout the simulation, only a few cells changed their wet and dry (WD) status. A stronger wave front is required to perform the water surge analysis. Although only a few cells changed their WD status, it was

sufficient to check whether Delft3D produces any wiggles in the solution. As expected, Delft3D did not introduce any wiggles into the solution.

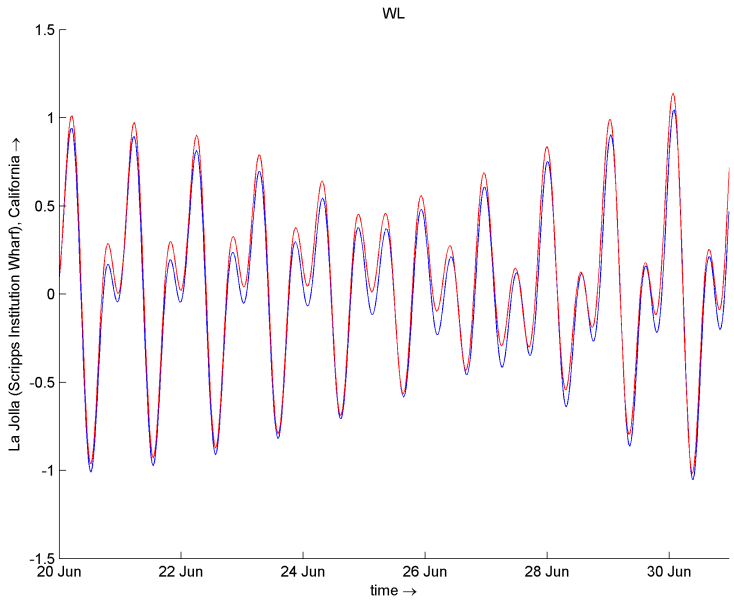

Figure 4.2. Changes in the water level at the La Jolla station. Delft3D output (blue line) and station value (red line).

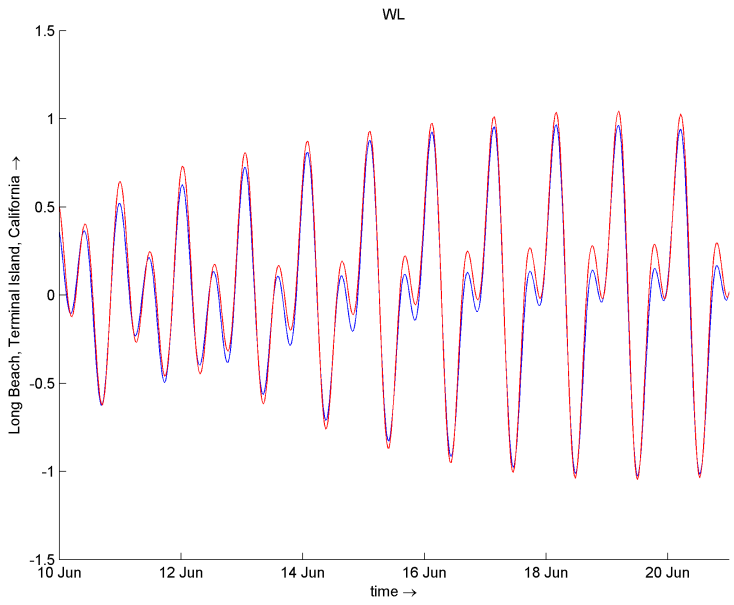

Figure 4.3. Changes in the water level at the Long Beach station. Delft3D output (blue line) and station value (red line).

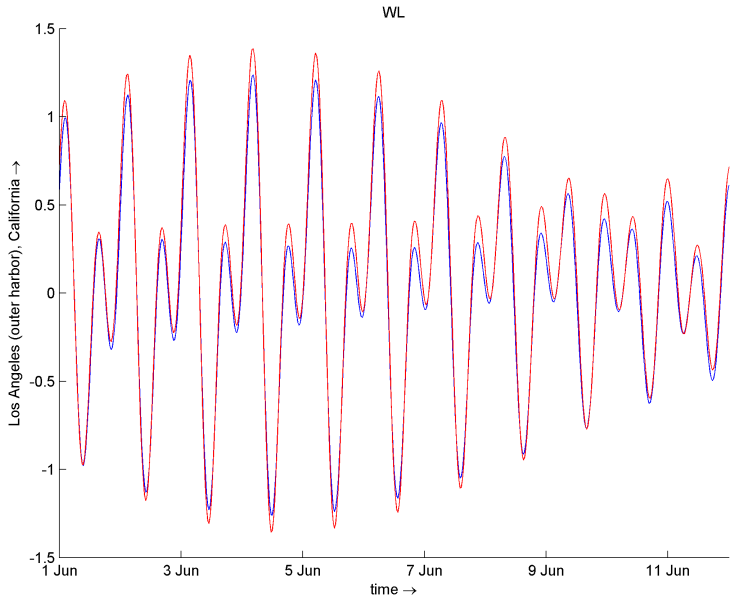

Figure 4.4. Changes in the water level at the Los Angeles station. Delft3D output (blue line) and station value (red line).

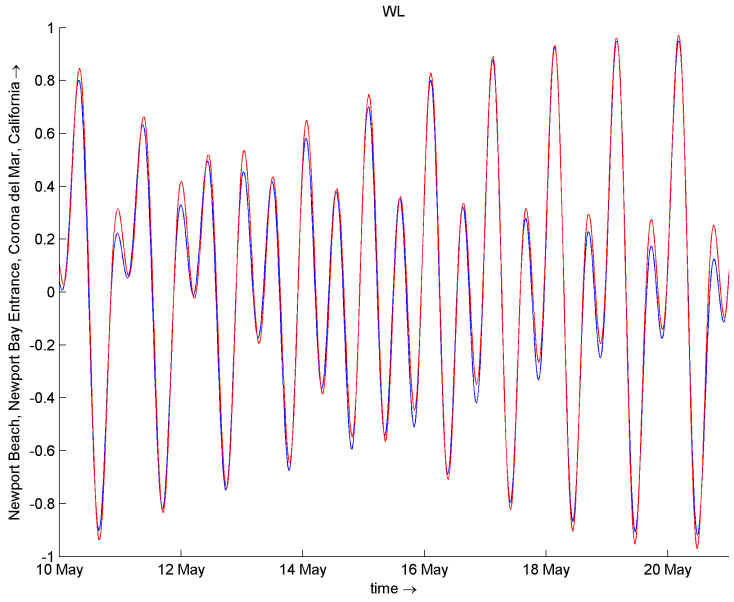

Figure 4.5. Changes in the water level at the Newport station. Delft3D output (blue line) and station value (red line).

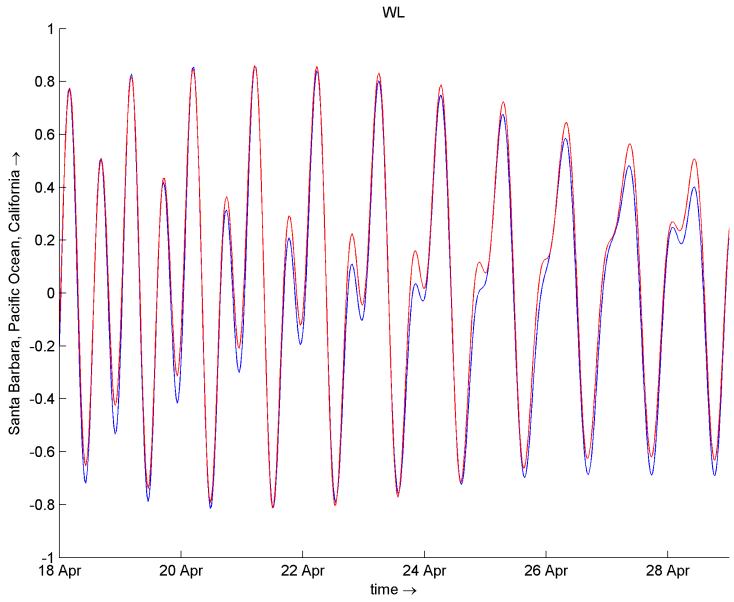

Figure 4.6. Changes in the water level at the Santa Barbara station. Delft3D output (blue line) and station value (red line).

Figure 4.7, Figure 4.8, Figure 4.9, and Figure 4.10 show a time series of how the velocity changes in Southern California. Throughout the day, except around noon, the dominant direction of the velocity is from north to south, as expected for Southern California. It should be noted that the model is only forced with tidal forces at the boundary and no wind forcing has been set.

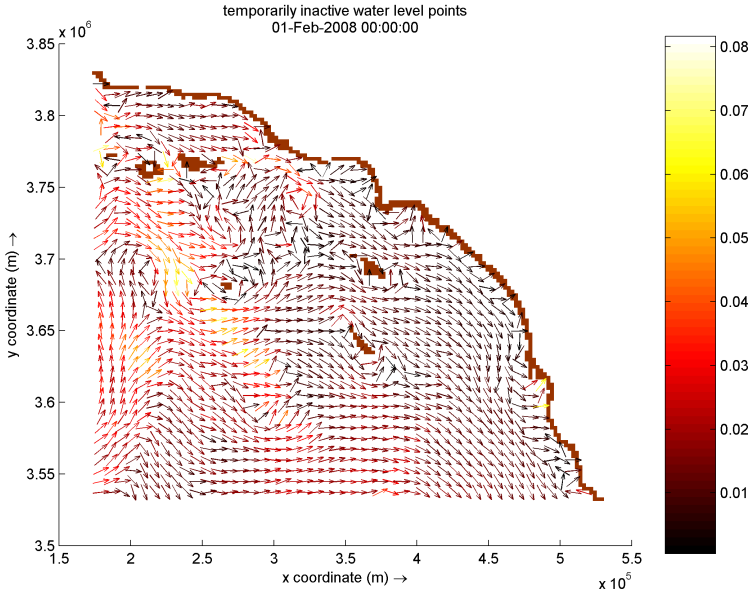

Figure 4.7. Velocity vectors in Southern California coarse model on Feb. 1, 2008 at 00:00.

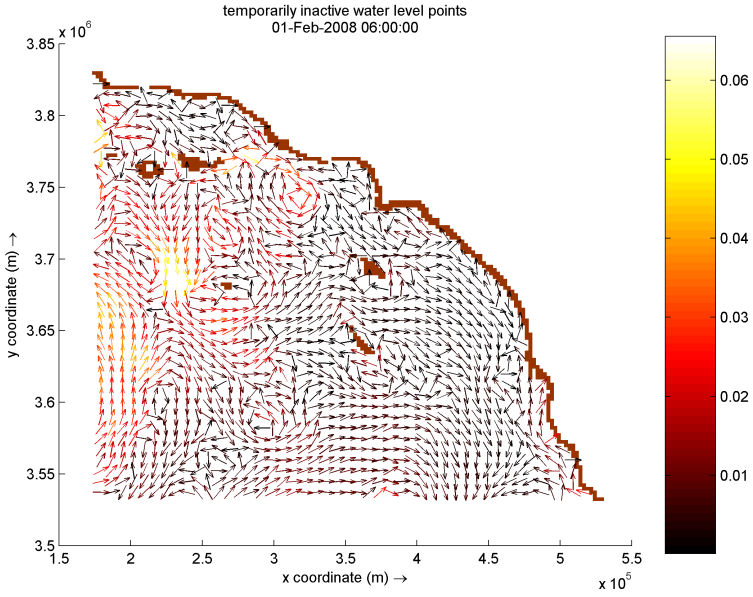

Figure 4.8. Velocity vectors in Southern California coarse model on Feb. 1, 2008 at 06:00.

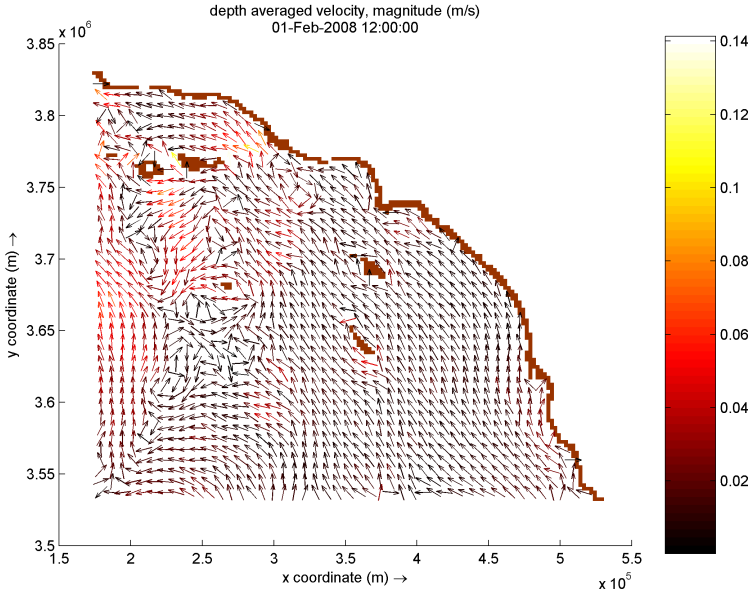

Figure 4.9. Velocity vectors in Southern California coarse model on Feb. 1, 2008 at 12:00.

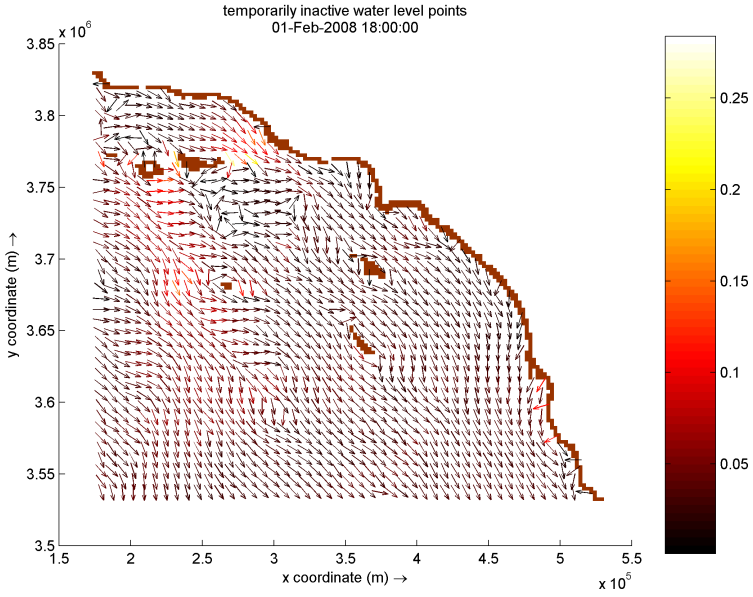

Figure 4.10. Velocity vectors in Southern California coarse model on Feb. 1, 2008 at 18:00.

4.2 $50\,m$ Resolution San Diego Bay Model

In previous section, the coarse single domain model was represented. This section focuses on the fine resolution single domain model, i.e. $50 \times 50\,m$. Again, first it is explained how the model was setup; followed by discussing the model output results.

4.2.1 Model Setup

The same coordinate system, i.e., the UTM zone 11N; and datum, i.e., the WGS84, were used for the high-resolution model. The domain was spatially discretized using $dx = dy = 50\,m$. Total of 522 grid points were selected in the M-direction (i.e., the eastward direction) and 402 grid points were selected in the N-direction (i.e., the northward direction). Therefore, the model covers an area of $26.1\,km \times 20.1\,km$ (Figure 4.11). Again, those parts of land with an elevation of $20\,m$ or higher were set to be always dry. Any parts of land with an elevation lower than $20\,m$ have the potential to become wet or dry. Only one layer in the vertical direction was selected, making it a true shallow water equation model. SDBathy, USGS SRTM, USGS Southern California, and GEBCO (for a total of four data sets) were combined to interpolate the bathymetry throughout the entire domain.

The start of the simulation time was set at January 1, 2008. The stable time step, using ADI scheme, was found to be 0.01 minute. One day of simulation took approximately two days and twenty hours to simulate. As a result, it was not possible to do any forecasting, and so it was decided to simulate only one day. The water level for the entire domain was set at $0\,m$ over the entire domain.

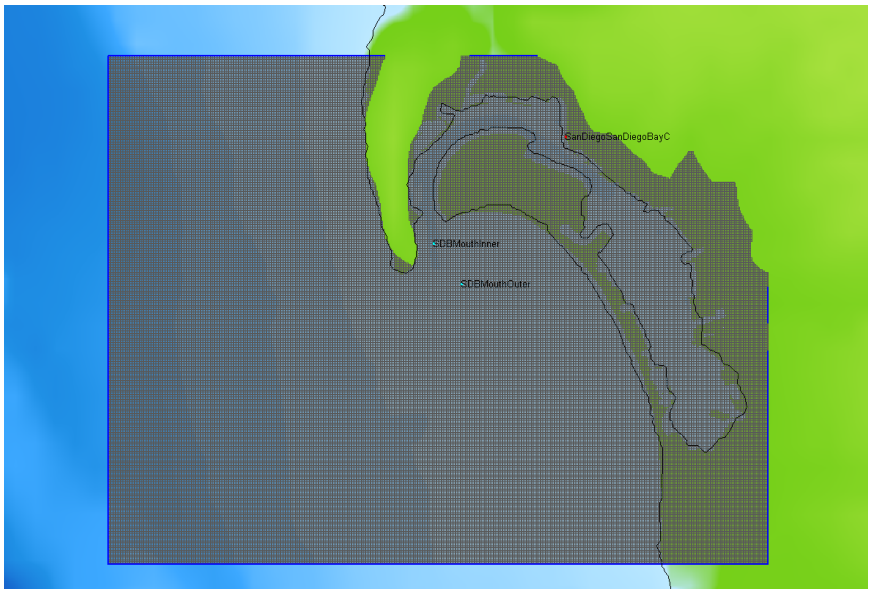

Figure 4.11. High-resolution San Diego Bay model.

Over land no flow condition is enforced. However, as in southern California model, astronomic boundary condition was used over open boundaries. The Earth gravity was set to $9.81\,m/s^2$ and the water density was set at $1024\,kg/m^3$. The Manning formulation was chosen for the bottom friction. Only one monitoring station was selected in this domain and that was the San Diego station. All other tide stations were located outside the model domain. However, two extra points close to the bay entrance were also selected. The history file was written every 30 minutes and the map file was set to be written every 6 hours; however, no restart file was selected.

4.2.2 Model Results

Before discussing any of the outputs, it should be noted that the model ran for only 1 day (which took 2 days and 20 hours of computer time); therefore, the model has not yet gone through the complete spin-up time. However, the water level profile during this period still matches relatively well with those of the San Diego Bay station (Figure 4.12). The velocity field is shown in Figure 4.13.

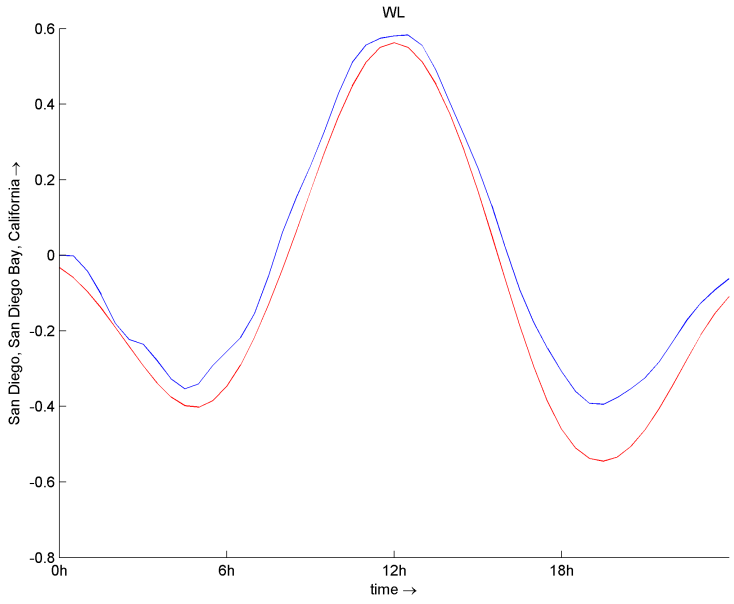

Figure 4.12. Changes in the water level during the first day of simulation. Delft3D output (blue line) and station value (red line).

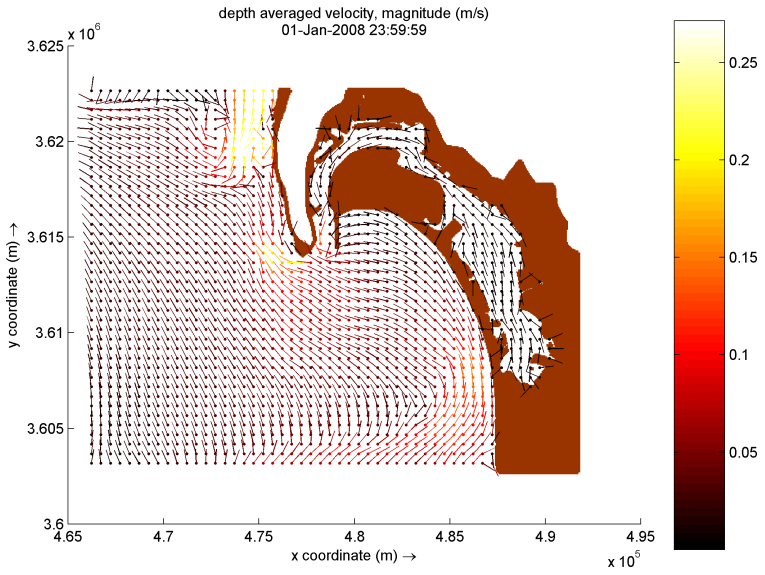

Figure 4.13. Velocity vectors in high-resolution San Diego Bay Model.

CHAPTER 5
NESTED SAN DIEGO BAY MODEL

In previous chapter two different single domain model were presented. Although, both single domain models have their own applications; it is recommended to generate a nested model, i.e. a fine resolution model nested inside a coarse resolution model. Nesting has many benefits, including a better and more realistic boundary conditions provided for fine resolution model. As before, first the model setup is discussed. Later, the model outputs are presented and discussed.

5.1 MODEL SETUP

To create a nested model, two models are needed: a coarse resolution model and a fine resolution model. For this project, it was decided to use the previously developed model with a $2.5\,km$ resolution as the coarse model. The second model, which is the high-resolution model, is then nested inside of the coarse resolution model. It was decided to keep the same resolution used in our previous high-resolution model (i.e., $dx = dy = 50\,m$). However, in order to speed up the model simulation time, we decided to decrease the total domain coverage. For the finer grid, a total of 399 grid points where selected in the M-direction (i.e., the eastward direction), and 331 grid points were selected in the N-direction (i.e., the northward direction). Therefore, the model covers an area of $19.95\,km \times 16.55\,km$. SDBathy, USGS Southern California, and GEBCO (for a total of three data sets) were combined to interpolate the bathymetry throughout the entire domain.

The simulation start time was set at January 1, 2008. The stable time step, using ADI scheme, was found to be 0.01 minute. Since the domain is much smaller, 2 days of simulation took only a bit more than 4 hours of computer time. It should be noted that most of this speed-up is actually due to using a stronger machine equipped with an Intel i3 processor with 16GB of random access memory (RAM). Nothing was changed for the coarser model. The water level for the entire domain was at $0\,m$ for the entire domain.

The boundary conditions for the coarser grid were kept the same. However, the boundary conditions for the finer model nested inside the coarser model were changed to a ”time Series.” A set of observation points along the boundary of the finer grid were created in the coarser grid model. After the coarser grid model was finished, the water level from the coarser grid was fed at the boundary of the finer model. This is only achievable by setting the boundary type to time series.

The Earth's gravity was set to $9.81\,m/s^2$ and the water density was set at $1024\,kg/m^3$. The Manning formulation was chosen for the bottom friction. Only one monitoring station was selected in the finer model, and that was the San Diego domain. The same monitoring points were kept in the coarse grid model. The history file was written every 30 minutes and the map file was set to be written every 24 hours; however, no restart file was selected.

5.2 Model Results

The coarse grid was run for 6 months again; but only to be able to extract the data every 30 minutes at the boundary of the finer model. Later, the finer model ran for 2 days, which took about 4 hours of computer time, and it was forced with the time series data provided by the coarse grid. The stable time step for the fine grid was 0.01 minutes; hence, the time series was linearly interpolated in time at each time step.

The model output for the water level at the San Diego Bay station is shown in Figure 5.1. It is clear that the model requires more spin-up time, and has not yet converged to the real solution. Figure 5.2 compares the output of the nested model versus the output of the single domain model for the San Diego Bay. It can be seen that both models follow the same trend; however, the single domain requires less spin-up time.

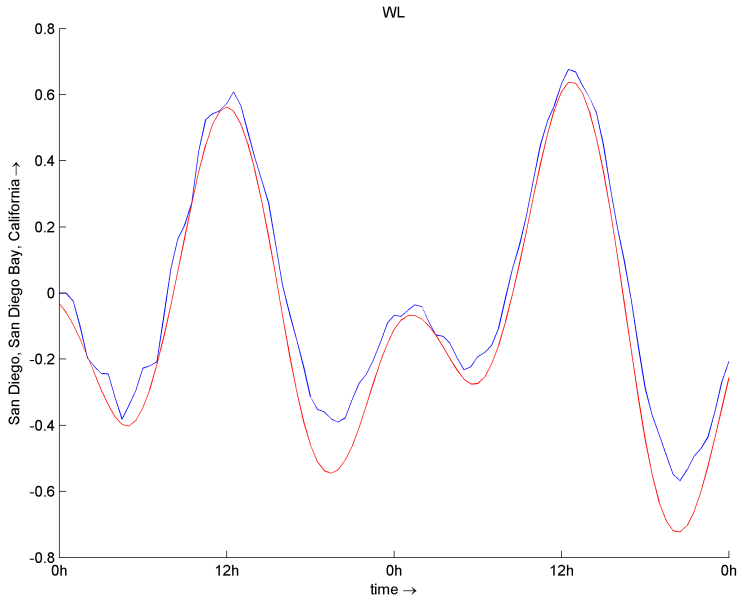

Figure 5.1. Changes in the water level at the San Diego bay station for the nested model during the first two days of the simulation. Delft3D output (blue line) and station value (red line).

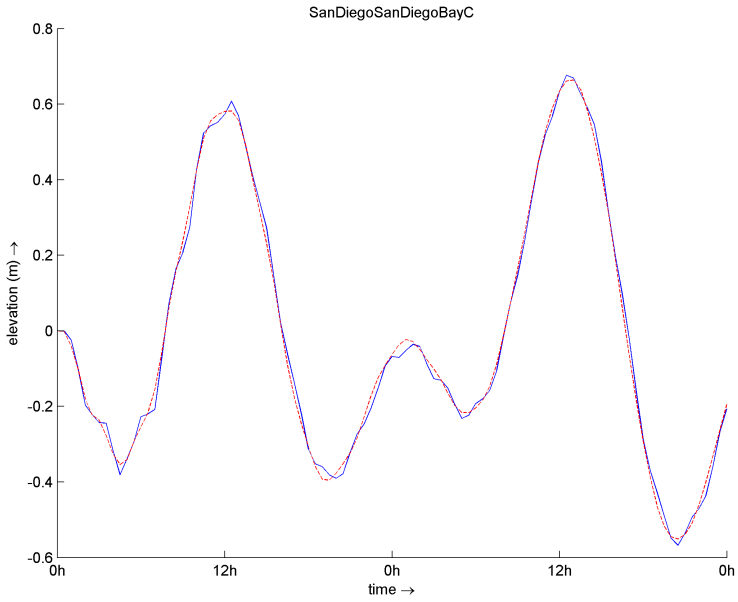

Figure 5.2. Changes in the water level at the San Diego Bay station. Nested San Diego Bay model (blue line) and single domain San Diego Bay model (dashed red line).

CHAPTER 6
CONCLUSION AND POSSIBLE FUTURE WORKS

Another hydrodynamic model of the San Diego Bay was developed in this project. Past modeling efforts of the region were mainly adapted using EFDC and CH3D models of San Diego Bay. However, they were all based on a curvilinear grid that was developed by the US Navy, known as the CH3D Navy grid. The CH3D grid, on average, provides a grid resolution of 100 m. None of these models allowed for wetting and drying; a process that is very important in defining the fate of certain important biological and ecological aspects of the region.

Unlike previous efforts, the model developed in this project provides high-resolution hydrodynamic data consistently for all areas of the San Diego Bay. The grid resolution was set to $50\,m$ throughout the entire domain, and all of Coronado Island and the San Diego Bay's surrounding areas were covered by the high-resolution mesh, making this model capable of performing wetting and drying studies for cases of severe conditions. More importantly, this model is based upon available free source codes and data sets, and can be executed on a regular desktop computer; unlike the Navy model, whose demonstration costs alone were over half a million dollars.

Two high-resolution models were developed of the San Diego Bay for this project: one that could act autonomously (single domain model), and one that needed to be nested inside a coarser model that covered a larger area (multi-domain or nested model). Both models were successful at estimating water levels; so, why build the more complicated, nested model? The single domain model is limited to study events and features restricted to or originating in the San Diego Bay only, assuming that all the boundary conditions are provided properly (which is not always possible); however, the multi-domain model (nested model) can be used to study the effects of events that occur beyond the San Diego region, and how they can affect the bay and its ecology. Since the detailed (or the fine resolution) model is nested inside a coarser model and located far from its boundaries, errors in boundary conditions of the coarser model will have less effect on the finer model. For example, if a certain pollutant is released into the waters around the Los Angeles port, would it end up in the San Diego Bay, considering that the main water flow along Southern Californian coasts is from north to south? The nested model is well suited to provide answers for these types of questions and modeling scenarios. A majority of previous works provided only a single domain; in fact, as

far as the author is aware, only one other nested model of the region exists (which is also based on Delft3D). However, the fate of that attempt is unknown, and no results have yet been made public, aside from a few report pages.

Some past attempts also tried to force the model using the available measurements only at the boundaries; however, these measurements were performed sparsely, and were used slightly off of the real locations. For example, in one case, measurements of La Jolla, were used to set the conditions at the San Diego Bay's entrance. Moreover, it appears that there is no study or model that uses data assimilation techniques in the San Diego Bay. Therefore, it is widely suggested to perform a simulation using data assimilation in the future. Still, the author believes that the quality of the results already attained (without any data assimilation) may demotivate the future researcher to go through the hurdles of modeling with data assimilation.

San Diego Bay is not large; thus, except in severe storm situations, wind stress plays little to no role in the hydrodynamics of the bay. Moreover, San Diego bay is well shielded by Coronado Island and its surrounding topography, which causes further reduction of the wind speed (and therefore, wind stress); making wind less effective inside of the bay. Due to certain structures, i.e., wave breakers at the bay's entrance (Figure 6.1), the bay is also relatively protected against high waves from the Pacific Ocean. However, it is still recommended to add wind stress as a forcing factor in the model, in order to determine whether this further reduces any error in the simulated water level. It should be noted that, though the results obtained in this project are within the range of acceptable error, due to the lack of time available it was decided not to include the force of the wind; moreover, it was not possible for the author to obtain high-resolution wind data for the model developed here.

Figure 6.1. The San Diego Bay's entrance and structures protecting the bay.

BIBLIOGRAPHY

[1] M. ABOUALI AND J. E. CASTILLO, *Unified curvilinear ocean atmosphere model (ucoam): A vertical velocity case study*, Mathematical and Computer Modelling, doi: 10.1016/j.mcm.2011.03.023 (2011).

[2] A. ADCROFT AND J. MARSHAL, *A new treatment of the coriolis terms in c-grid models at both high and low resolutions*, Monthly Weather Review, 127 (1999), pp. 1928–1936.

[3] J. M. BECKERS, H. BURCHARD, J. M. CAMPIN, E. DELEERSNIJDER, AND P. P. MATHIEU, *Another reason why simple discretizations of rotated diffusion operators cause problems in ocean models: Comments on "isoneutral diffusion in a z co-ordinate ocean model"*, American Meteorological Society, 28 (1998), pp. 1552–1559.

[4] T. BELL, *Lee waves in stratified flows with simple harmonic time dependence*, Journal of Fluid Mechanics, 67 (1975), pp. 705–722.

[5] J. BERNSTEN, *Internal pressure errors in sigma-coordinate ocean models*, Journal of Atmospheric and Oceanic Technology, 19 (2002), pp. 1403–1414.

[6] A. F. BLUMBERG AND G. L. MELLOR, *Modelling vertical and horizontal diffusivities with the sigma co-ordinate system*, Monthly Weather Review, 113 (1985), p. 1379.

[7] D. B. CHADWICK, I. RIVERA-DUARTE, G. ROSEN, P. WANG, R. C. SANTORE, A. C. RYAN, P. R. PAQUIN, S. D. HAFNER, AND W. CHOI, *Demonstration of an integrated compliance model for predicting copper fate and effects in dod harbors*, Project ER-0523, SPAWAR, 2008.

[8] T. CHRISTIANSEN, P. WIBERG, AND T. MILLIGAN, *Flow and sediment transport of a tidal salt marsh surface*, Estuarine, Coastal and Shelf Science, 50 (2000), pp. 315–331.

[9] B. CUSHMAN-ROISIN, *Introduction to Geophysical Fluid Dynamics*, Prentice-Hall, Engelwood Cliffs, 1994.

[10] DELTARES, *Delft3D-FLOW User Manual*, Delft, the Netherlands, 2011.

[11] ——, *Delft3D Installation Manual*, Delft, the Netherlands, 2011.

[12] C. ECKART, *Properties of water, part 2. the equation of state of water and sea water at low temperatures and pressures*, American Journal of Science, 256 (1958), pp. 225–240.

[13] R. M. GERSBERG, *San diego bay terrain model progress report*, technical report, San Diego State University, 2012.

[14] S. GILLE, M. YALE, AND D. SANDWELL, *Global correlation of mesoscale ocean variability with sea floor roughness from satellite altimetry*, Geophysical Research Letters, 27 (2000), pp. 1251–1254.

[15] S. GRIFFIES, C. BOENING, F. BRYAN, E. CHASSIGNET, R. GERDES, H. HASUMI, A. HIRST, A. M. TREGUIER, AND D. WEBB, *Development in ocean climate modelling*, Ocean Modelling, 2 (2000), pp. 123–192.

[16] R. HANEY, *On the pressure gradient force over steep topography in sigma coordinate ocean models*, Journal of Physical Oceanography, 21 (1991), pp. 610–619.

[17] I. JAMES, *Modeling pollution dispersion, the ecosystem and water quality in coastal waters: A review*, Environmental Modeling and Software, 17 (2002), pp. 363–385.

[18] S. KARICKHOFF, S. DAVID, AND A. TRUDY, *Sorption of hydrophobic pollutants on natural sediments*, Water Research, 13 (1979), pp. 241–248.

[19] K. KRETTENAUER, *Numerische simulation turbulenter konvektion ueber gewelten flaechen*, PhD thesis, DLR, Oberpfaffenhofen, Germany, 1991.

[20] K. KRETTENAUER AND U. SCHUMANN, *Numerical simulation of turbilent convection over wavy terrain*, Journal of Fluid Mechanics, 237 (1992), pp. 261–299.

[21] L. LAURENT AND A. THURNHERR, *Intense mixing of lower thermocline water on the crest of the mid-atlantic ridge*, Nature, 448 (2007), pp. 680–683.

[22] J. J. LEENDERTSE, *Aspects of a computational model for long-period water-wave propagation*, rm-5294-rr, Rand Corporation, Santa Monica, 1967.

[23] J. J. LEENDERTSE, *Turbulence modelling of surface water flow and transport: part 4a*, Journal of Hydraulic Engineering, 114 (1990), pp. 603–606.

[24] J. J. LEENDERTSE, R. C. ALEXANDER, AND S. K. LIU, *A three-dimensional model for estuaries and coastal seas.*, technical report, Rand Corporation, Santa Monica, CA, 1973.

[25] J. J. LEENDERTSE AND E. C. GRITTON, *A water quality simulation model for well mixed estuaries and coastal seas*, Technical Report Vol. 2 Computation Procedures R-708-NYC, Rand Corporation, Santa Monica, CA, 1971.

[26] U. LUMBORG AND A. WINDELIN, *Hydrography and cohesive sediment modeling: application to the romo dyb tidal area*, Journal of Marine Systems, 38 (2003), pp. 287–303.

[27] J. L. MARTIN AND S. MCCUTCHEON, *Hydrodynamics and Transport for Water Quality Modeling*, Lewis Publishers, 1999.

[28] J. MCCALPIN, *A comparison of second-order and fourth-order pressure gradient algorithms in a sigma coordinate ocean models*, International Journal Numerical Methods in Fluids, 18 (1994), pp. 361–383.

[29] E. MCDONALD AND R. T. CHENG, *Issues related to modeling the transport of suspended sediments in northern san francisco bay, california*, in 3rd International Conference on Estuarine and Coastal Modeling., 1994, pp. 1–41.

[30] T. J. PEELING, *A proximate biological survey of san diego bay, california*, Technical Report 389, Naval Undersea Center, San Diego, California, 1975.

[31] J. PENG AND E. Y. ZENG, *An integrated geochemical and hydrodunamic model for tidal coastal environments*, Marine Chemistry, 103 (2007), pp. 15–29.

[32] N. A. PHILLIPS, *A coordinate system having some special advantages for numerical*

forecasting, Journal of Meteorology, 14 (1957), pp. 184–185.

[33] K. POLZIN, J. TOOLE, J. LEDWELL, AND R. SCHMITT, *Spatial variability of turbulent mixing in the abyssal ocean*, Science, 276 (1997), pp. 93–96.

[34] W. RODI, *Turbulence models and their application in hydraulics*, in IAHR Paper presented by the IAHR-Section on Fundamentals of Division 2: Experimental and Mathematical Fluid Dynamics, 1984.

[35] G. S. STELLING, *On the construction of computational methods for shallow water flow problems*, Technical Report 35, TUDelft, Delft, The Netherlands, 1984.

[36] G. S. STELLING AND S. P. A. DUINMEIJER, *A staggered conservative scheme for every froude number in rapidly varied shallow water flows*, International Journal Numerical Methods in Fluids, 43 (2003), pp. 1329–1354.

[37] G. S. STELLING AND J. J. LEENDERTSE, *Approximation of convective processes by cyclic aoi methods*, in Estuarine and coastal modeling, Proceedings 2nd Conference on Estuarine and Coastal Modelling, M. L. Spaulding, K. Bedford, and A. Blumberg, eds., Tampa, 1992, ASCE, pp. 771–882.

[38] G. S. STELLING AND J. A. T. M. VAN KESTER, *On the approximation of horizontal gradients in sigma co-ordinates for bathymetry with steep bottom slopes*, International Journal Numerical Methods in Fluids, 18 (1994).

[39] TETRATECH, *Receiving water model configuration and evaluation for san diego bay toxic pollutants tmdls*, technical report, Tetra Tech, Inc., San Diego, CA, 2008.

[40] A. THURNHERR AND K. RICHARDS, *Hydrography and high-temperature heat flux of the rainbow hydrothermal site (mid-atlantic ridge)*, Journal of Geophysical Research, 106 (2001), pp. 9411–9426.

[41] A. THURNHERR, K. RICHARDS, C. GERMAN, G. LANE-SERFF, AND K. SPEER, *Flow and mixing in the rift valley of the mid-atlantic ridge*, Journal of Physical Oceanography, 32 (2002), pp. 1763–1778.

[42] A. THURNHERR AND K. SPEER, *Boundary mixing and topography blocking on the mid-atlantic ridge in the south atlantic*, Journal of Physical Oceanography, 33 (2003), pp. 848–862.

[43] UNESCO, *Background papers and supporting data on the international equation of state 1980*, Technical Report 38, UNESCO, 1981.

[44] UNESCO, *The practical salinity scale 1978 and the international equation of state of seawater 1980*, Tenth report of the Joint Panel on Oceanographic Tables and Standards 36, UNESCO, 1981.

[45] P. F. WANG, R. T. CHENG, K. RICHTER, E. S. GROSS, D. SUTTON, AND J. W. GARTNER, *Modeling tidal hydrodynamics of san diego bay, california*, Journal of the American Water Resources Association, 34 (1998), pp. 1123–1140.